In the Claims:

Claims 24-27 are withdrawn. Applicants reserve the right to file a divisional application for claims 24-27.

1. (Original) A processor, comprising:

a selection unit, having a first input to receive a thread conditions signal indicative of one or more execution stalls, the selection unit selecting a thread from a plurality of threads based on the thread conditions signal; and

a selection mux, having a first input coupled to an output of the selection unit to receive the thread selection signal and a second input to receive a plurality of decoded instructions, the selection mux outputting a decoded instruction from the plurality of instructions in response to the thread selection signal.

2. (Original) The processor of claim 1, wherein the selection unit selects a thread each clock cycle.

3. (Original) The processor of claim 1, wherein the selection unit further comprises:

a high priority unit, having a first input coupled to the first input of the selection unit, the high priority unit selecting a thread associated with a high priority instruction from the plurality of threads;

a low priority unit, having a first input coupled to the first input of the selection unit, the low priority unit selecting a thread with a low priority instruction from the plurality of threads; and

a control mux, having a first input coupled to an output of the high priority unit and having a second input coupled to an output of the low priority unit, the control mux selecting between the high priority selection and the low priority selection.

4. (Original) The processor of claim 3, wherein a second input to the high priority unit and a second input to the low priority unit are coupled to receive a previous thread

selection, and the high priority unit and the low priority unit use round robin arbitration to select a next thread.

5. (Original) The processor of claim 3, wherein the high priority unit has a third input and the low priority unit has a third input, the third high priority unit input and the third low priority unit input receiving an instruction conditions signal indicative of availability, the high priority unit selecting between threads with an available high priority instruction and the low priority unit selecting between threads with an available low priority instruction.

6. (Original) The processor of claim 3, wherein the control mux defaults to select the high priority selection.

7. (Original) The processor of claim 3, wherein the control mux selects the low priority selection after a predetermined number of successive high priority selections.

8. (Original) The processor of claim 3, wherein the thread conditions signal comprises a global stall indicator, wherein the control mux holds a selected thread responsive to an active global stall indicator.

9. (Original) The processor of claim 3, wherein the control mux selects the low priority selection if there is no high priority selection.

10. (Original) The processor of claim 3, wherein the control mux repeats a previous selection.

11. (Original) The processor of claim 1, wherein the thread conditions signal is indicative of thread eligibility, and wherein the selection unit selects between eligible threads.

12. (Original) The processor of claim 1, wherein the execution stall comprises a data cache miss.

13. (Original) The processor of claim 1, wherein the execution stall comprises an external resource stall.

14. (Original) The processor of claim 1, wherein the execution stall comprises an interlock.

15. (Original) The processor of claim 1, wherein the execution stall comprises a memory operation ordering.

16. (Original) The processor of claim 1, wherein the selection unit has a second input to receive an instruction conditions signal, the selection unit selecting a thread based on the instruction conditions signal.

17. (Original) The processor of claim 16, wherein the instruction conditions signal is indicative of one or more thread ages.

18. (Original) The processor of claim 16, wherein the instruction conditions signal is indicative of whether an instruction is available for each of the plurality of threads, and wherein the selection unit selects between available threads.

19. (Original) The processor of claim 16, wherein the instruction conditions signal comprises a priority indicator generated external to the selection unit to indicate a priority level of one or more threads.

20. (Original) A network processing device to process a plurality of network packets, comprising:

a plurality of packet processing instructions;

a plurality of threads comprising a stream of one or more packet processing instructions from the plurality of packet processing instructions, each thread corresponding to a network packet from the plurality of network packets; and

a multithreaded processor, comprising an input to receive the one or more packet processing instructions based on an instruction fetch sequence, and to execute the one or more instructions based on an execution dispatch sequence.

21. (Original) A network processor having a multithreaded pipeline, comprising:

an upper pipeline, having an input coupled to receive a signal indicative of an instruction queue depth corresponding to a plurality of threads, the upper pipeline determining an instruction fetch sequence for the plurality of threads based on the instruction queue depth signal; and

a lower pipeline, comprising a first input to receive decoded instructions and a second input to receive a thread conditions signal, the lower pipeline determining a thread execution sequence based on the thread conditions signal, the thread conditions signal indicative of an execution stall corresponding to the plurality of threads.

22. (Original) The network processor of claim 21, wherein the lower pipeline determines the thread execution sequence independent of the instruction fetch sequence.

23. (Original) The network processor of claim 21, wherein the thread execution sequence comprises a sequence of instructions that are resequenced relative to the instruction fetch sequence.

24. (Withdrawn) The network processor of claim 21, wherein the upper pipeline comprises an instruction unit to fetch the instructions.

25. (Withdrawn) The network processor of claim 21, wherein the upper pipeline comprises a decode unit to decode the instructions.

26. (Withdrawn) The network processor of claim 25, wherein the decode unit further comprises an instruction queue, an input of the instruction queue coupled to receive the decoded instructions from an output of the decode unit, the decode unit storing decoded instructions.

27. (Withdrawn) The network processor of claim 21, wherein the upper pipeline comprises:

an instruction unit to fetch instructions according to the instruction fetch sequence;

a decode unit, coupled to an output of the instruction unit, to decode fetched instructions, the decode unit comprising an instruction queue to store decoded instructions; and

an instruction queue depth signal line, coupled to an output of the instruction queue and an input of the instruction unit, the instruction queue depth signal line to feed back an indication of an instruction queue depth.

28. (Original) The network processor of claim 21, wherein the lower pipeline comprises:

a thread interleaver, having an input coupled to the first input of the lower pipeline, to dispatch a decoded instruction corresponding to a thread according to the thread execution sequence.

29. (Original) The network processor of claim 28, wherein the thread interleaver determines the thread execution sequence using two-level round robin arbitration.

30. (Original) The network processor of claim 21, wherein the lower pipeline comprises an execution unit, wherein the execution switches execution from a first thread to a second thread independent of an execution stall associated with the first thread.

31. (Original) The network processor of claim 21, wherein the lower pipeline comprises an execution unit, wherein the execution unit executes a different thread each clock cycle.

32. (Original) The network processor of claim 21, wherein the lower pipeline comprises a thread interleaver directly coupled to an execution pipeline.

33. (Original) The network processor of claim 21, wherein the lower pipeline determines the thread execution sequence based on a two-level round robin arbitration.

34. (Original) A method of thread interleaving, comprising:

receiving a plurality of decoded instructions associated with a plurality of threads;

receiving thread conditions associated with the plurality of threads, the thread conditions indicative of an execution stall;

selecting a thread from the plurality of threads based on the thread conditions; and

outputting a decoded instruction corresponding to the thread selection.

35. (Original) The method of claim 34, wherein the selecting comprises selecting a thread each clock cycle.

36. (Original) The method of claim 34, further comprising:

executing the decoded instruction associated with a different thread each clock cycle, wherein the selecting the thread occurs each clock cycle

37. (Original) The method of claim 34, further comprising:

executing a first decoded instruction associated with a first thread during a first clock cycle; and

executing a second decoded instruction associated with a second thread during a second clock cycle, the executing the second decoded instruction independent of a thread condition associated with the first thread.

38. (Original) The method of claim 34, further comprising:

attempting to execute the selected instruction;

detecting an execution stall during the attempting to execute; and

generating an execution stall signal.

39. (Original) The method of claim 34, wherein the selecting one of the plurality of threads further comprises:

selecting a thread associated with a high priority instruction from the plurality of threads;

selecting a thread associated with a low priority instruction from the plurality of threads; and

selecting between the high priority selection and the low priority selection.

40. (Original) The method of claim 39, further comprising:

receiving a previous thread selection, wherein the high priority selection and the low priority selection use round robin arbitration to select a next thread.

41. (Original) The method of claim 39, further comprising:

receiving instruction conditions indicative of availability, wherein the high priority selection comprises selecting a thread associated with an available high

priority instruction, and wherein the low priority selecting comprises selecting a thread associated with an available low priority instruction.

42. (Original) The method of claim 39, wherein the selecting between the high priority selection and the low priority selection comprises selecting by default the high priority selection.

43. (Original) The method of claim 39, wherein the selecting between the high priority selection and the low priority selection comprises selecting the low priority selection after a predetermined number of successive high priority selections.

44. (Original) The method of claim 39, further comprising:

receiving a global stall indicator, wherein the selecting between the high priority selection and the low priority selection comprises holding a previous selection responsive to an active global stall indicator.

45. (Original) The method of claim 39, wherein the selecting between the high priority selection and the low priority selection comprises selecting the low priority selection if there is no high priority selection.

46. (Original) The method of claim 34, wherein the execution stall comprises a data cache miss.

47. (Original) The method of claim 34, wherein the execution stall comprises an external resource stall.

48. (Original) The method of claim 34, wherein the execution stall comprises an interlock.

49. (Original) The method of claim 34, wherein the execution stall comprises a memory operation ordering.

50. (Original) The method of claim 34, wherein the execution stall a thread age.

51. (Original) The method of claim 34, wherein the selecting is based on an external priority indicator.

52. (Original) A processor, comprising:

queuing means for receiving a plurality of decoded instructions associated with a plurality of threads;

selection means for receiving thread conditions associated with the plurality of threads, the thread conditions indicative of an execution stall, and selecting a thread from the plurality of threads based on the thread condition; and

outputting means, coupled to the queuing means and the selection means, for outputting a decoded instruction from the queuing means corresponding to the thread selection.

53. (Original) The processor of claim 52, wherein the selection means selects a thread each clock cycle.

54. (Original) The processor of claim 52, wherein the selection means further comprises:

high priority selection means for selecting a thread associated with a high priority instruction from the plurality of threads;

low priority selection means for selecting a thread associated with a low priority instruction from the plurality of threads; and

control means for selecting between the high priority selection and the low priority selection.

55. (Original) The processor of claim 54, wherein the high priority means and the low priority means receive a previous selection, the high priority means and the low priority means using round robin arbitration to select a next thread.

56. (Original) The processor of claim 54, wherein the high priority means and the low priority means receive instruction conditions indicating availability, the high priority means selecting a thread associated with an available high priority instruction, and the low priority means selecting a thread associated with an available low priority instruction.

57. (Original) The processor of claim 54, wherein the control means defaults to the high priority thread selection.

58. (Original) The processor of claim 54, wherein the control means selects the selected low priority thread after a predetermined number of successive high priority thread selections.

59. (Original) The processor of claim 54, wherein the control means receives a global stall indicator, the control means holding a previous selection responsive to an active global stall indicator.

60. (Original) The processor of claim 54, wherein the control means selects the low priority selection if there is no high priority selection.

61. (Original) The processor of claim 52, wherein the execution stall comprises a data cache miss.

62. (Original) The processor of claim 52, wherein the execution stall comprises an external resource stall.

63. (Original) The processor of claim 52, wherein the execution stall comprises an interlock.

64. (Original) The processor of claim 52, wherein the execution stall comprises a memory operation ordering.

65. (Original) The processor of claim 52, wherein the selection means receives a priority indicator generated external to the selection means.

66. (Original) A method of thread interleaving, comprising:

determining an upper pipeline instruction fetch sequence based on a per thread instruction queue depth information; and

determining a lower pipeline thread execution sequence based on a thread condition indicative of an execution stall.

67. (Original) A processor, comprising:

upper pipeline means for determining an instruction fetch sequence based on per thread instruction queue depth information; and

lower pipeline means, coupled to the upper pipeline means, for determining a pipeline thread execution sequence based on a thread condition indicative of an execution stall.

68. (Original) A computer product comprising:

a computer-readable medium having computer program instructions and data embodied thereon for thread interleaving, comprising:

receiving a plurality of decoded instructions associated with a plurality of threads;

receiving thread conditions associated with the plurality of threads, the thread conditions indicative of an execution stall;

selecting a thread from the plurality of threads based on the thread conditions; and

outputting a decoded instruction corresponding to the selected thread.